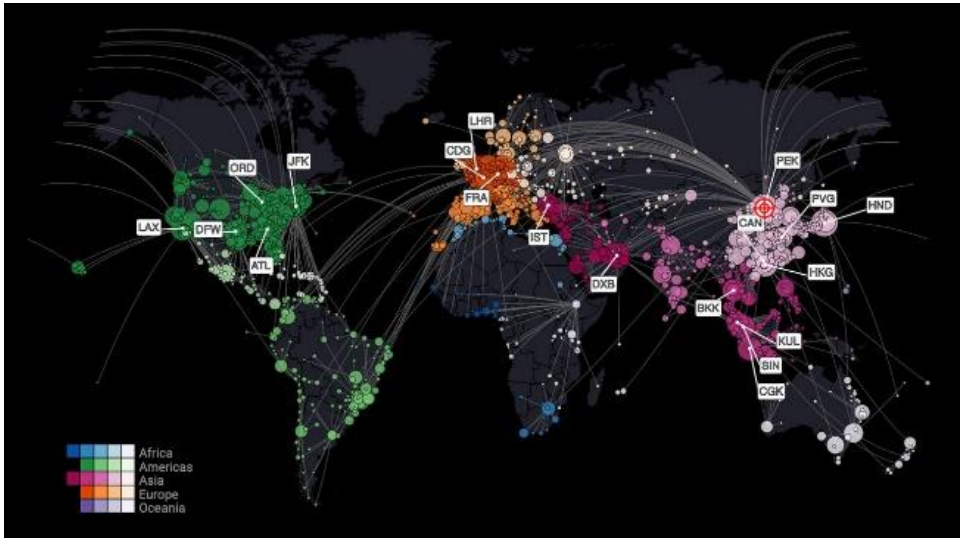


How Reliable are Pandemic Models?

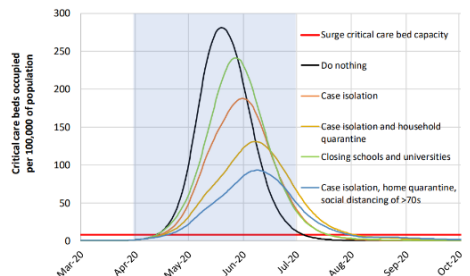
An urgent proposal to improve pandemic modelling



Government policies to manage the COVID-19 pandemic affect the safety, well-being and economy of the UK. Public policy decisions are informed from the use pandemic modelling. How reliable are pandemic models? How dependable are the data and the computer programs that underpin them? Professor Harold Thimbleby discusses his concerns on current practices, and presents an urgent call to action to the software safety community to help address the issues he raises.

Pandemic modelling

Government planning for responding to the COVID-19 pandemic has been based on epidemic modelling. Typically, epidemic modelling tries to make predictions of infections and death rates from mathematical models operating on medical and national data, such as demographics, social habits, and what is known about the infective agent. Of course, computers are essential to manage the data, simulate



The quality of current pandemic models is not open to scientific scrutiny. The models are either not peer reviewed, or have been peer reviewed very casually.

or solve the models, and to present results. The UK's socially disruptive and economically costly isolation, is based on epidemiological models suggesting that the NHS would be overwhelmed without it.

It was a surprise to discover that the influential Imperial College model [1], which had a central role in early COVID-19 planning, was developed from an undocumented C program [2]. There is a chain of references starting from [1] showing the "same" program was in use for different diseases and different countries, in a research programme going back at least 15 years earlier [3]. A very old, undocumented program written in C cannot be reliably modified.

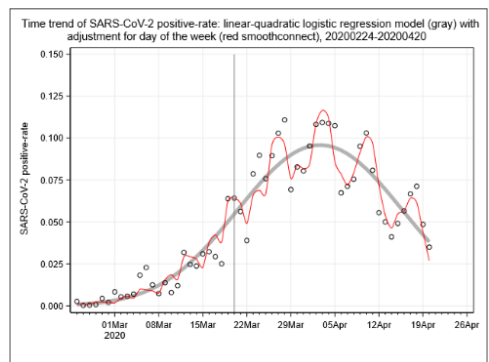
The epidemiological modelling literature ignores programming, and how hard good programming reliably is.

A classic epidemiology review paper [4] says, "we use the words 'computational modelling' loosely," and then, curiously, exclusively discusses mathematical modelling, as if it is unaware that programming is an essential part of modelling, and no less complex than the mathematics. Without good software engineering — including correct data management, version control, appropriate numerical methods, dependable programming, etc — the models cannot be used to reliably inform public policy.

A recent systematic review [5] of published COVID models, covering individual diagnosis and prognosis in clinical care, including apps and online tools, *completely* ignores model implementation. Apps don't work unless they have working programs! It should be noted that flowcharts and other implementations of models, which the review considered, must be designed as carefully as computer programs: they *are* programs, but intended for direct human use (which raises human factors issues).

Very few epidemiological papers provide any description of their software, let alone access to any software. Those that do, provide access to essentially undocumented code, that is, code with trivial comments.

In short: despite its critical nature, epidemiological modelling used to inform public policy has a very naïve attitude to software. The quality of the current pandemic models is not open to scientific scrutiny, and implies that the models are either not peer reviewed, or have been peer reviewed very casually.



Why professional software engineering is required

Playing the piano is easy if you do not worry about quality, but playing the piano well is very difficult. We can all hear the difference when a wrong piano key is pressed, or is pressed out of time, but programming is different.

A mistake in a program may not be noticed for a very long time, if ever. It is therefore very easy for programmers to over-estimate their programming skills because they cannot see their errors — in fact, poor programmers are less able to find their own errors, and, in a vicious cycle, they therefore further over-estimate their skills.

It seems likely that epidemiology research labs give programming tasks to people who do not realise how incompetent they are at programming; indeed, probably nobody else in their lab recognises the problems with their programming. It seems to work, so they think it must be working correctly.

Software Engineering is the discipline of programming well, and is therefore the approach that epidemiologists should be taking when developing computer models.

A quick summary of Software Engineering would cover at least the following:

- Define requirements: how reliable does code have to be at what cost?
- Correct by construction (formal methods etc)
- Using dependable programming languages
- Tool use
- Defensive programming
- Version control, repositories, configuration, open source
- Rigorous testing
- Good documentation
- Usability (at least: demonstrate independent people can use the code)
- Reusing quality-assured solutions
- Simplicity
- Explicit compliance with appropriate standards
- Effective teamwork to cover the skills required.

This list is merely an outline curriculum. Sir Tony Hoare FRS noted as far back as 1980 [6] that “in any respectable branch of engineering, failure to observe such elementary precautions would have long been against the law.”

The point is: the relevant skills for turning epidemic modelling into reliable software to use those models cannot be learned quickly, even if one has the aptitude. In particular, epidemiologists alone cannot be expected to have the relevant skills.

Getting professional software engineering to where it matters

Unless professional software engineering practices are used, computer-based models, and epidemiological models in particular, and results from running them, must be considered unreliable, and unreliable in unknown ways. Unfortunately, results from poor computer models have been published and acted on without adequate scrutiny of the code in use. The accepted *laissez faire* approach is inadequate for use in any modelling informing national policies or clinical treatment.

This raises the question of how to improve.

Professional Software Engineering is essential, but how can it be applied or made available? Given the evident absence of mature software engineering awareness and skills in leading epidemiology research, what can be done?

An analogous situation arises in ethics. There are some sorts of experiments and methods that are ethically unacceptable, but few people have the ethical expertise to make mature judgements — particularly when it comes to assessing their own work. Misuse of data, exploiting vulnerable people, and not obtaining consent are typical problems. Universities, research labs, national funders, and others therefore require Ethics Boards, who review the ethical quality of proposed research. Medical journals will not publish research that has not had appropriate ethical review.

Journals should start requiring not just “reproducibility” but actual evidence of independent reproduction of results, particularly for pandemic modelling.

Quality software engineering requires an explicit proactive approach [7], including compliance monitoring. Establishing Software Engineering Boards seems the best way to make rapid improvements in software quality, without imposing onerous processes and succumbing to “the best is the enemy of the good” pitfalls.

The Software Engineering Boards would authorise as well as provide advice to guide the detailed implementation of high-quality programming. Active, senior, professors of software engineering should be on these Boards; this is not a job for people who are not qualified in the area and not actively connected

with the true state of the art. There are many high-quality software companies (especially those specialising in safety-critical areas like aviation) who would be willing to help. Many university departments of computer science have qualified and experienced software engineers who can also help.

It is important to note that merely making code open source or otherwise available for scrutiny (for instance, providing it in Supplementary Information with papers) is *not* sufficient to ensure quality. Undocumented, badly-written code is inscrutable, even if its original programmers think otherwise. In fact, making poorly developed code available merely enables others to reproduce and propagate the poor work.

Software Engineering Boards must be used to ensure that critical code is of high standard, and must provide assurances for scientific papers, Government reports, and so on, that the software is of appropriate quality. Only then is it worth sharing and using more widely.

Note that an essential feature of good science is *reproducibility* — indeed, if results cannot be reproduced, what has anything contributed? When a modelling paper presents results from a model, it is very important to reproduce those results *without using the same code*. Better still, research should be reproduced without sharing libraries or APIs (for example, results from a model using, say, the language R, might be reproduced in a different language, say, Mathematica). Reproducing the same results relying on the same codebase tells you little. The more independent reproductions of results the greater the evidence for belief in the implications. Given the practical difficulties of applying professional software engineering, scientific journals should immediately start encouraging reproduction of prior results, particularly for pandemic modelling where research insights affect national and world populations.

In most science it is straight forward to ensure results are “reproducible”: enough details have to be provided for other researchers to reproduce the results. In software, this is not sufficient. Merely providing the code and data does not mean that others can reproduce the results. The code may not be portable, the code may require specific versions of compilers,

specific hardware, and so on. All the code might be there, but the configuration details may be missing. And so on. We need higher standards for code *if it is trying to make a scientific contribution*: it needs to *actually* be reproduced by independent parties before we can be sure it is truly reproducible. Software Engineering Boards could confirm that independent researchers have reproduced the results claimed. Conversely, the “Software statement” in a paper could just say no independent party has confirmed the results.

Just as medical papers today require conflicts of interest statements, data availability statements, and ethics board clearance, we must move to papers being required to include Software Engineering Board clearance statements. Like Ethics Boards, Software Engineering Boards might become, or be perceived as becoming, onerous and heavy handed. It is essential that Software Engineering Boards have (and perhaps are chaired by) experienced, professional software engineers to avoid this problem.

Unlike Ethics Boards, which provide hands-off oversight, Software Engineering Boards should provide professional advice where necessary, perhaps providing training or be hands-on actually helping develop appropriately reliable software. This is not free, so clearly all research, particularly medical research, should be required to include adequate funding for professional software engineering.

Can it be done?

It is striking how the NHS has very rapidly changed everyday practice to use remote consultation because COVID-19 has put pressure on it to continue work without risking cross-infecting staff and patients. Likewise, COVID-19 should have put pressure on modelling papers informing public policy to be rigorous, given the seriousness of the choices facing Government.

Therefore, as a matter of national priority, Software Engineering Boards could be rapidly established to provide direct access to mature software engineering expertise for both researchers and for journals seeking competent peer reviewers. In addition, particularly during a pandemic or other emergency, as now, Software Engineering Boards would provide direct access to their expertise for Governments and public policy organisations.

In the longer run, it is likely that a professional organisation (such as the UK’s Royal Academy of Engineering) would provide a professional framework for Software Engineering Boards. In particular, until there are national qualifications, nobody — certainly nobody without professional training in software — really knows just how bad they are at software engineering.

It is hoped that this article encourages grass roots and organisational responses to the problems identified.

Conclusions

Current epidemiological practice is an alarming basis for public policy.

Mature software engineering methodologies are essential for reliable modelling, including all critical programming. We must prioritise getting appropriate professional software engineering skills and resources to bear on the COVID-19 pandemic without delay. We should also be planning to help improve modelling to better manage any future crises. (it’s sobering to recall that austerity was inspired by easily avoidable bugs in an Excel spreadsheet [8]).

Since mature software engineering skills are not widely available, Software Engineering Boards, analogous to Ethics Boards, could be rapidly established to provide the necessary up-to-date knowledge and oversight for critical projects (like pandemic modelling) to succeed. Software Engineering Boards promise an effective way out of the amateurish use of computers in all research.

Indeed, epidemic modelling is not the only area affected by poor programming, and Software Engineering Boards would have far-reaching benefits across all fields. Elsewhere [9] we have shown how unreliable software undermines the safety of medical systems more generally, putting patients at risk. Current medical device regulation does not address these problems — arguably, Software Engineering Boards should have a broad remit to identify and address software problems anywhere.

References

- [1] N. M. Ferguson, D. Laydon, G. Nedjati-Gilani, *et al*, “Impact of non-pharmaceutical interventions (NPIs) to reduce COVID-19 mortality and healthcare demand,” <http://www.imperial.ac.uk/media/imperial-college/medicine/sph/ide/gida-fellowships/Imperial-College-COVID19-NPI-modelling-16-03-2020.pdf>, 16 March 2020, accessed 22/04/2020
- [2] N. M. Ferguson, http://twitter.com/neil_ferguson/status/1241835454707699713, accessed 22/04/2020.
- [3] N. M. Ferguson, D. A. T. Cummings, C. Fraser, J. C. Cajka, P. C. Cooley & D. S. Burke, “Strategies for mitigating an influenza pandemic,” *Nature* 2005; 437:209–214. DOI 10.1038/nature04017
- [4] H. Heesterbeek, R. M. Anderson, V. Andreasen, S. Bansal, D. De Angelis, C. Dye, K. T. D. Eames, W. J. Edmunds, S. D. W. Frost, S. Funk, T. D. Hollingsworth, T. House, V. Isham, P. Klepac, J. Lessler, J. O. Lloyd-Smith, C. J. E. Metcalf, D. Mollison, L. Pellis, J. R. C. Pulliam, M. G. Roberts, C. Viboud & Isaac Newton Institute IDD Collaboration, “Modeling infectious disease dynamics in the complex landscape of global health,” *Science* 2015; 347(6227):aaa4339. DOI 10.1126/science.aaa4339
- [5] L. Wynants, B. van Calster, M. M. J. Bonten, *et al*, “Prediction models for diagnosis and prognosis of covid-19 infection: Systematic review and critical appraisal,” *BMJ* 2020; 369:m1328. DOI 10.1136/bmj.m1328
- [6] C. A. R. Hoare, “The Emperor's Old Clothes,” *Communications of the ACM* 1981; 24(2):75–83. DOI 10.1145/358549.358561
- [7] B. Shneiderman, “Opinion: The dangers of faulty, biased, or malicious algorithms requires independent oversight,” 2016; 113 (48)13538–13540. DOI 10.1073/pnas.1618211113
- [8] T. Herndon, M. Ash & R. Pollin, “Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff,” *Cambridge Journal of Economics*, 38(2):257–279, 2014. DOI 10.1093/cje/bet075
- [9] H. Thimbleby, *Fix IT: How to solve the problems of digital healthcare*, Oxford University Press, 2020.

Harold Thimbleby

Harold Thimbleby is See Change Fellow in Digital Health at Swansea University, Wales. He has recently completed *Fix IT: How to solve the problems of digital healthcare*, which will be published by Oxford University Press later in 2020. Harold won the BCS Wilkes Medal and has previously been a Royal Society Wolfson Research Merit Award Holder. He has published over 300 refereed papers, and has given over 600 presentations around the world.

Image Attributions

global map: Dirk Brockmann

graph: reproduced from [1] under CC BY-NC-ND 4.0 license

time trend: Hscherb / CC BY-SA (<https://creativecommons.org/licenses/by-sa/4.0>)